REMARKS

The present application was filed on February 8, 2000 with claims 1-25. In a response filed April 15, 2005, Applicants added independent claim 26. Claims 1, 13, 25 and 26 are now the independent claims.

In the outstanding Office Action, the Examiner rejected claims 1-26 under 35 U.S.C. §102(b) as being anticipated by U.S. Patent No. 5,768,510 to Gish (hereinafter "Gish").

In this response, Applicants amend the independent claims and respectfully traverse the §102(b) rejection of claims 1-26.

Regarding the §102(b) rejection of claims 1-26, the Office Action contends that Gish discloses all of the claim limitations recited in the subject claims. Applicants respectfully assert that Gish fails to teach or suggest all of the limitations in claims 1-26, for at least the reasons presented below.

It is well-established law that a claim is anticipated only if each and every element as set forth in the claim is found, either expressly or inherently described, in a single prior art reference. *Verdegaal Bros. v. Union Oil Co. of California*, 814 F.2d 628, 631, 2 U.S.P.Q.2d 1051, 1053 (Fed. Cir. 1987). Applicants assert that the rejection based on Gish does not meet this basic legal requirement, as will be explained below.

While Applicants believe that the claims, prior to this response, are patentably distinguishable over Gish, Applicants have nonetheless amended independent claims 1, 13, 25 and 26 to further define the invention in a sincere effort to expedite the application through to issuance.

Thus, the claimed invention, as recited for example in amended independent claim 1, provides a method for use in a client/server system of reducing interactions between a client and server in association with an application being accessed by the client at the server. The method comprises the steps of: configuring the server to store a model associated with the application and to execute view-generating and controller logic associated with the application; and configuring the client to store at least a subset of the model associated with the application and to execute at least a subset of the view-generating and controller logic associated with the application, wherein one or more portions of the application are performed at the client without the client having to interact with the server, and further wherein the client and the server both execute the respective model and view-

generating and controller logic resident thereon. The underlined language represents the added claim language. Independent claims 13, 25 and 26 recite similar limitations and have been amended in a similar manner.

As explained at page 3, line 22-27, of the present specification: "[t]he invention addresses performance by employing a dual-MVC approach, in which a subset of the application's Model-View-Controller reside and execute on the client, and the full Model-View-Controller and View-Generating-Logic reside and execute on the server, thereby reducing the number of required server interactions." FIG. 3 of the present application illustrates such an inventive dual-MVC approach.

Thus, the claimed invention recites that the server "store[s] a model associated with the application and execute[s] view-generating and controller logic associated with the application," and the client "store[s] at least a subset of the model associated with the application and execute[s] at least a subset of the view-generating and controller logic associated with the application" such that "the client and the server both execute the respective model and view-generating and controller logic resident thereon."

The Abstract of Gish explains that disclosed therein is:

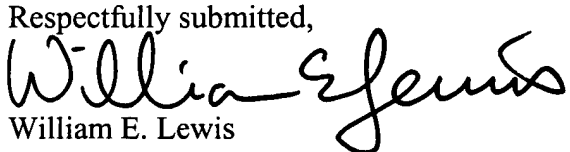
> An interprise [sic] computing manager in which an application is composed of a client (front end) program which communicates utilizing a network with a server (back end) program. The client and server programs are loosely coupled and exchange information using the network. The client program is composed of a User Interface (UI) and an object-oriented framework (Presentation Engine (PE) framework). The UI exchanges data messages with the framework. The framework is designed to handle two types of messages: (1) from the UI, and (2) from the server (back end) program via the network. The framework includes a component, the mediator which manages messages coming into and going out of the framework. The system includes software for a client computer, a server computer and a network for connecting the client computer to the server computer which utilize an execution framework code segment configured to couple the server computer and the client computer via the network, by a plurality of client computer code segments resident on the server, each for transmission over the network to a client computer to initiate coupling; and a plurality of server computer code segments resident on the server which execute on the server in response to initiation of coupling via the network with a particular client utilizing the transmitted client computer code segment for communicating via a particular communication protocol.

Thus, while Gish discloses "a client (front-end) program" and a "server (backend) program," and that "server computer code segments resident on the server . . . execute[s] on the server in response to initiation of coupling via the network with a particular client," this is not the same as the claimed feature of "configuring the server to store a model associated with the application and to execute view-generating and controller logic associated with the application, configuring the client to store at least a subset of the model associated with the application and to execute at least a subset of the view-generating and controller logic associated with the application . . . wherein the client and the server both execute the respective model and view-generating and controller logic resident thereon.

That is, the claimed invention recites that the client and the server both execute the respective model and view-generating and controller logic resident thereon (dual MVC approach). In other words, in the claimed approach, both the client and the server execute the same model and view-generating and controller logic associated with the application (albeit, with the client executing at least a subset thereof). On the other hand,Gish discloses a client-server system in which the client and server are running different logic, as distinguished by "front end" and "back end" in FIG. 5 of Gish. Further, Gish does not execute view-generating logic on the server. Gish allows the client to download the front-end code from the server, but the front-end code is never active (i.e., never executed) when it is on the server (i.e., it is stored inactively), see, column 18, lines 14-59.

Accordingly, Applicants assert that independent claims 1, 13, 25 and 26, as well as the claims which depend therefrom, are patentable over Gish and therefore allowable. Such dependent claims also recite patentable subject matter in their own right. Withdrawal of the §102(b) rejection of claims 1-26 is therefore respectfully requested.

Respectfully submitted,



Date: October 24, 2005

William E. Lewis
Attorney for Applicant(s)
Reg. No. 39,274
Ryan, Mason & Lewis, LLP
90 Forest Avenue
Locust Valley, NY 11560
(516) 759-2946